Basket Level Recommendation Using Association Rule Mining

Technical Report

May 17, 2025

Abstract

The report describes a memory-efficient market basket analysis system which generates product recommendations for retail customers through association rule mining. The system discovers customer purchase patterns by analyzing which items tend to co-appear in transaction records. The system provides two recommendation methods which include association rule-based suggestions using lift scores and category-based suggestions for products outside the current customer basket. The analysis produced 248 association rules which included support and confidence and lift metrics. The method delivers exceptional value for checkout recommendation systems and cross-selling enhancement applications. The system maintains memory efficiency which enables it to handle extensive retail data sets.

1 Data Overview

The implementation utilizes transaction data with the following key attributes:

- **TransactionID**: Unique identifier for each purchase transaction
- **CustomerID**: Unique identifier for each customer
- **ProductCode**: Unique identifier for each product
- **ItemName**: Product description/name
- **Category**: Product category classification
- **DateTime**: Timestamp of the transaction
- **Quantity**: Number of units purchased

Data statistics from the implementation:

- Total transactions: 29,408
- Unique customers: 1,000
- Unique products: 19,779
- Product categories: 10
- Total rows: 477,906

2 Methodology

2.1 Association Rule Mining Approach

Association rule mining discovers interesting relationships between items in large transactional datasets. The implementation follows these key steps:

2.1.1 Transaction Basket Creation

- Grouping products by transaction ID to create customer baskets
- Limiting basket size to a maximum number of unique items for memory efficiency

2.1.2 Frequent Itemset Discovery

- Identifying products that appear in transactions more than a minimum threshold
- Efficiently calculating item frequencies while minimizing memory usage
- Filtering to keep only items that meet the support threshold

2.1.3 Frequent Pair Generation

- Discovering pairs of products that frequently appear together in transactions
- Implementing a memory-efficient approach by processing one basket at a time
- Applying minimum support thresholds to filter relevant pairs

2.1.4 Association Rule Creation

- Generating rules in the form "if customer buys X, they might also buy Y"
- Calculating key metrics for each rule:
  - Support: Proportion of transactions containing both items
  - Confidence: Probability of Y given X (conditional probability)
  - Lift: Ratio of observed support to expected support if items were independent

2.2 Recommendation Strategies

The system implements two complementary recommendation approaches:

2.2.1 Association Rule-Based Recommendations

- Using discovered rules to recommend products for items in the basket
- Aggregating lift scores when multiple basket items suggest the same recommendation
- Excluding items already in the basket from recommendations
- Ranking products by their aggregated lift scores

2.2.2 Category-Based Recommendations

- Identifying product categories not represented in the current basket
- Recommending the most popular products from missing categories
- Ensuring recommendation diversity across the product catalog
- Complementing association rules when insufficient recommendations are available

3 Implementation Details

The implementation is designed for memory efficiency and practical application in retail environments. Key components include:

3.1 Memory-Efficient Basket Analysis

```
def memory_efficient_basket_analysis(df, min_support_count=10,
    min_confidence=0.1, max_basket_items=50):
    """
    Perform memory-efficient basket analysis for product recommendations
        .

    Parameters:
    -----------
    df : pandas DataFrame
        Transaction data with TransactionID and ProductCode columns
    min_support_count : int
        Minimum number of baskets an itemset must appear in
    min_confidence : float
        Minimum confidence for an association rule
    max_basket_items : int
        Maximum number of unique items to consider in a basket (for
            memory efficiency)

    Returns:
    --------
    rules_df : pandas DataFrame
        Association rules with antecedent, consequent, support,
            confidence, and lift
    """
    # Implementation details (see code in notebook)
```

This function efficiently implements the apriori-like algorithm for finding frequent itemsets and generating association rules while minimizing memory usage:

- Limits basket size to control memory consumption
- Filters items by frequency before generating pairs
- Processes one basket at a time rather than creating large intermediate data structures
- Calculates metrics in a streaming fashion

3.2 Recommendation Functions

```
def recommend_products_for_basket(basket, rules_df, frequent_items,
    product_map, top_n=5):
    """Recommend products based on current basket items."""
    # Implementation aggregates scores from multiple rules
    # and returns top recommendations
```

```
def recommend_by_category(basket, df, product_map, top_n=1):
    """Recommend top products from categories not in the basket."""
    # Implementation identifies missing categories
    # and recommends popular products from them
```

3.3 Production-Ready Integration

The implementation includes a practical checkout recommendation function that combines both recommendation strategies:

```
def get_recommendations_for_checkout(basket_items):
    """Get recommendations for a given basket at checkout."""
    # Get association-based recommendations
    assoc_recommendations = recommend_products_for_basket(
        basket_items, rules_df, frequent_items, product_map, top_n=3)

    # If we don't have enough association recommendations, add category-based ones
    if len(assoc_recommendations) < 3:
        cat_recommendations = recommend_by_category(
            basket_items, df, product_map, top_n=3-len(
                assoc_recommendations))

        # Combine both types of recommendations
        all_recommendations = [
            f"{product} (Score: {score:.2f})" for product, score in
                assoc_recommendations
        ] + [
            f"{product} (from {category})" for category, product in
                cat_recommendations
        ]
    else:
        all_recommendations = [
            f"{product} (Score: {score:.2f})" for product, score in
                assoc_recommendations
        ]

    return all_recommendations
```

4 Results and Evaluation

4.1 Generated Association Rules

The implementation successfully generated 248 association rules with a minimum support count of 10 transactions. The top rules sorted by lift score show interesting product affinities:

Antecedent	Consequent	Support	Confidence	Lift
Helga's Sourdough Bread	Coles Thin & Crispy Pizza Bases	0.0004	0.0769	13.63
Coles Thin & Crispy Pizza Bases	Helga's Sourdough Bread	0.0004	0.0723	13.63
Simson's Pantry Low Carb Spinach	Mission Crust Plain Pizza	0.0003	0.0625	10.56
Mission Crust Plain Pizza	Simson's Pantry Low Carb Spinach	0.0003	0.0575	10.56
Coles Bakery Salted Caramel Scrolls	Coles Bakery Super Soft Hot Dog Rolls	0.0003	0.0546	9.68

Table 1: Top Association Rules by Lift Score

These rules reveal meaningful product associations, such as:

- Complementary pizza ingredients (bases and toppings)

- Bread and bakery product affinities
- Related prepared food items

4.2 Sample Recommendations

For a test basket containing:

- La Fournee Doree Sliced Brioche Loaf Plain | 500g
- Grinders Rich & Bold Espresso Ground Coffee | 200g
- Provedore Salami Tasting Plate | 100g

The system recommended:

1. Olay Luminous Niacinamide Super Serum | 30mL (from Health & Beauty)
2. Birds Eye Frozen Seasoned Sides Mediterranea ... 600g (from Frozen)
3. Coles Organic Brushed Potatoes Prepacked | 2kg (from Fruit & Vegetables)

These recommendations demonstrate the category-based approach, suggesting products from categories not represented in the original basket.

4.3 Performance Metrics

The implementation demonstrated efficient resource usage:

- Analysis completed in 9.24 seconds for a dataset of 477,906 transactions
- Memory consumption controlled by limiting basket size to 50 items
- Adaptive parameter selection (trying different support thresholds)
- Scalable to large retail datasets

5 Limitations and Challenges

5.1 Data Sparsity

- The support values for top rules are quite low (0.0003-0.0004), indicating relative rarity of co-occurrences
- Minimum support threshold needed to be carefully tuned to generate sufficient rules
- Many potential product combinations do not appear frequently enough to form rules

5.2 Cold Start Problem

- New products without sufficient transaction history cannot participate in rule generation
- Requires fallback strategies like category-based recommendations
- Limited effectiveness for highly specialized or niche products

5.3 Scalability Considerations

- Current implementation limits basket size for memory efficiency (max_basket_items=50)
- Processing very large transaction datasets may require further optimization
- Association rule storage and retrieval efficiency becomes important at scale

5.4 Rule Quality

- Rules with high lift but low support may represent coincidental rather than causal relationships
- Balancing between rule coverage and rule quality requires careful parameter tuning
- Current implementation does not handle seasonality or temporal patterns

6 Conclusion

The memory-efficient market basket analysis implementation successfully identifies important product relationships while creating suitable recommendations for retail consumers. The system uses association rule-based and category-based recommendations to create a strong method for handling the natural sparsity found in retail transaction data.

The system shows practical use in retail environments through its ability to generate 248 association rules while maintaining efficient processing times. The recommendations cover multiple product categories while showing unexpected connections between different products.

The system fulfills essential business needs through its ability to generate personalized cross-selling recommendations and improve product discovery between categories and deliver real-time suggestions during checkout. The system's effectiveness and business value would increase through improvements in rule mining algorithms and temporal dynamics and personalization capabilities.